

# **CPP05 – Ex00 : Bureaucrat**

## **Fiche de Soutenance — Présentation & Explications Didactiques**

Cette fiche de soutenance est conçue pour vous aider à présenter clairement et professionnellement l'exercice CPP05 Ex00. Elle inclut une introduction, une explication progressive, des démonstrations concrètes et des réponses types aux questions fréquentes de l'examinateur.

## 1. Introduction générale

L'exercice Ex00 introduit la Programmation Orientée Objet en C++. Il s'agit de créer une classe simple mais rigoureusement encadrée : **Bureaucrat**. Ce premier exercice sert de fondation pour comprendre les concepts essentiels : gestion des erreurs avec exceptions, intégrité des données, surcharge d'opérateur et forme canonique des classes.

L'objectif est d'apprendre à structurer une classe en C++ selon des règles strictes, tout en produisant un code sécurisé, lisible et conforme au standard C++98.

## 2. Objectifs pédagogiques de l'exercice

- Créer une classe robuste avec attributs privés
- Appliquer les règles de validation (grade entre 1 et 150)
- Manipuler les exceptions héritant de `std::exception`
- Utiliser les blocs `try/catch` correctement
- Respecter la forme canonique C++
- Surcharger l'opérateur `<<` pour rendre l'objet affichable

## 3. Explication de la classe Bureaucrat

La classe Bureaucrat possède deux attributs :

- Un **nom constant**
- Un **grade** compris entre **1** (meilleur) et **150** (pire)

Si un grade invalide est fourni au constructeur ou lors d'une modification, la classe lance une exception dédiée :

- `GradeTooHighException`
- `GradeTooLowException`

## 4. Démonstrations concrètes

Création correcte :

```
Bureaucrat a("Alice", 42); std::cout << a;
```

Création incorrecte (exception) :

```
try { Bureaucrat b("Bob", 0); } catch (std::exception &e) { std::cout << e.what() << std::endl; }
```

Modification de grade :

```
Bureaucrat c("Charlie", 2); c.incrementGrade(); // 2 -> 1
```

Débordement (exception) :

```
Bureaucrat d("David", 1); d.incrementGrade(); // Erreur : grade trop haut
```

## 5. Surcharge de l'opérateur <<

La surcharge de l'opérateur d'insertion permet d'afficher un Bureaucrat comme une phrase lisible :

```
std::cout << a;
```

Format attendu : , bureaucrat grade .

## 6. La forme canonique

La classe doit respecter les 4 éléments de la forme canonique : - Constructeur par défaut - Constructeur de copie - Opérateur d'affectation - Destructeur

## 7. Questions fréquentes de l'examinateur

### Q1. Pourquoi le nom est-il constant ?

Parce qu'un bureaucrate ne change pas d'identité après sa création.

### Q2. Pourquoi incrementGrade() diminue la valeur du grade ?

Parce que 1 est le meilleur grade. Donc diminuer la valeur améliore le grade.

### Q3. Pourquoi utiliser des exceptions ?

Pour empêcher l'objet d'entrer dans un état invalide et garantir sa sécurité.

### Q4. À quoi sert operator<< ?

À rendre la classe facile à afficher et plus intuitive à utiliser.

## 8. Conclusion

Cet exercice pose les bases de la POO en C++. Il enseigne la structure d'une classe, la gestion professionnelle des erreurs, la protection de l'intégrité interne des objets et l'écriture d'un code lisible et conforme aux règles du module CPP05.